TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

VILLE TAHVANAINEN

KOTLININ SUOSION KASVU JA SEN SYYT

Kandidaatintyö

Tarkastaja: Sampo Suonsyrjä

TIIVISTELMÄ

VILLE TAHVANAINEN: Kotlinin suosion kasvu ja sen syyt
Kotlin's rise to fame and the reasons behind it
Tampereen teknillinen yliopisto
Kandidaatintyö, 17 sivua
Joulukuu 2018
Tietotekniikan kandidaatin tutkinto-ohjelma
Pääaine: Ohjelmistotekniikka
Tarkastaja: Sampo Suonsyrjä

Avainsanat: Kotlin, suosio, Android, Java

Tässä työssä perehdyttiin ohjelmointikieli Kotliniin ja vertailtiin sitä Javaan. Työn tarkoituksena oli selvittää Kotlinin suosion kasvua ja kasvun syitä. Lisäksi tarkasteltiin myös missä Kotlinia käytetään ja missä ei.

Ensin perehdyttiin hieman aihepiiriin, sekä Javan että Kotlinin perusteisiin. Perusteiden jälkeen käytiin läpi Kotlinin uusia ominaisuuksia, ja vertailtiin niitä Javaan. Sen jälkeen tutkittiin Kotlinin suosion kasvua tarkastellen käyttäjämääriä, käyttäjätyytyväisyyttä, kielen herättämää kiinnostusta, internetissä saatavilla olevan materiaalin määrää ja kysyntää työmarkkinoilla. Lopuksi pohdittiin suosion syitä.

Kotlin tarjoaa paljon uusia moderneja ominaisuuksia Javaan verrattuna, joita ohjelmoijat arvostavat, kuten tyhjäarvotarkastukset. Sen suosio on kasvanut voimakkaasti etenkin sen jälkeen, kun se sai statuksen virallisena Android-kielenä, ja Android onkin sen yleisin käyttötarkoitus. Puutteet työkalujen yhteensopivuudessa ovat yleisimpiä teknisiä syitä, jotka estivät Kotlinin käytön. Javan yleisyyden myötä Kotlinin potentiaalinen käyttäjäkunta on valtava, mutta ainakin työmarkkinoilla kielen kysyntä on edelleen verrattain vähäistä.

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

API	engl. Application programming interface, ohjelmointirajapinta
ART	engl. Android Runtime, suoritusympäristö
HTTP	engl. Hypertext Transfer Protocol, tiedonsiirtoprotokolla
IDE	engl. Integrated development environment, ohjelmointiympäristö
JDK	engl. Java development kit
JVM	engl. Java Virtual Machine, Java-virtuaalikone
MVC	engl. Model-view-controller, ohjelmiston arkkitehtuurimalli
SDK	engl. Software development kit
WORA	engl. write once, run anywhere, suunnittelufilosofia

1. JOHDANTO

Palveluiden ja sisällön siirtyessä entistä enenevissä määrin verkkoon, ja käytön tapahtuessa yhä useammin mobiililaitteen kautta, on myös näille alustoille tehtävän ohjelmistokehityksen määrä noususuhdanteessa.

Java on yksi maailman suosituimmista ohjelmointikielistä [7, 18]. Se käännetään tavukoodiksi, jota ajetaan erityisessä ympäristössä, Java-virtuaalikoneessa, joka tunnetaan myös lyhenteellä JVM. Virtuaalikoneen käyttö on yksi Javan vahvuuksista, ja se mahdollistaa tavukoodin alustariippumattomuuden. Virtuaalikoneen käytöllä siis abstrahoidaan käytettävä laitteisto tavukoodilta mahdollistaen saman tavukoodin suorittamisen alustasta riippumatta. [15] Muun muassa maailman suosituin älypuhelinkäyttöjärjestelmä [13] Android hyödyntää Java-ympäristöä ja kirjastoja. Myös muita ohjelmointikieliä voidaan kääntää Java-tavukoodiksi, ja näin hyödyntää jo olemassa olevia JVM-toteutuksia eri alustoille.

Kotlin on JetBrainsin Pietarissa kehittämä 2010-lukulainen moderni JVM-kieli. Kieli on nopeasti kasvattanut suosiotaan, ja vuonna 2017 Google nosti sen yhdeksi virallisista Android-kehityksessä tuetuista kielistä [16]. Tämän työn tarkoituksena on selvittää, miten ja miksi Kotlinin suosio ohjelmointikielenä on viime vuosien aikana kasvanut. Kieltä vertaillaan Javaan, jolla on edelleen vahva [12] asema sekä mobiili- että palvelinohjelmoinnissa ja etsitään syitä kielen nauttimalle suosiolle tai sille, miksi kieltä ei jossain tapauksessa käytetä. Suosion mittarina käytetään kieltä käyttävien kehittäjien määrää, kehittäjien tyytyväisyyttä, kielen herättämää kiinnostusta, internetissä saatavilla olevan materiaalin määrää ja kysyntää työmarkkinoilla.

Toisessa luvussa perehdymme Javaan, joka on vaikuttanut vahvasti Kotlinin suunnitteluun ja johon sitä usein verrataan. Kolmannessa luvussa tutustutaan Kotliniin ja sen ominaisuuksiin, sekä vertaillaan sitä Javaan. Neljännessä luvussa käydään läpi Kotlinin suosion kasvua erilaisilla mittareilla tarkasteltuna ja syitä havaitulle kehitykselle. Viidennessä luvussa vedetään tulokset yhteen.

2. JAVA

Java on alun perin Sun Microsystemsin kehittämä, nykyisin Oracle Corporationin omistama, luokkapohjainen, yleiskäyttöiseksi suunniteltu olio-ohjelmointikieli. Kielen kehitys aloitettiin vuonna 1991, ja sitä suunniteltiin käytettäväksi kulutuselektroniikassa, jossa käytetty ohjelmointikieli oli usein sidottu käytettävään kulloinkin käytettävään prosessoriin. Java ratkaisi siirrettävyysongelman virtuaalikoneen ja tavukoodin yhdistelmällä. Java-koodi käännetään ensin laitteistoriippumattomaksi tavukoodiksi, jonka jälkeen sitä voidaan suorittaa missä tahansa laitteessa, kunhan sille löytyy Java-virtuaalikoneen toteutus. [3]

2.1 Perusteet

Javan suunnittelulähtökohtia olivat yksinkertaisuus, alustariippumattomuus, siirrettävyys, monisäikeisyys, dynaamisuus, turvallisuus ja oliokeskeisyys [3]. Javan syntaksi pohjautuu paljolti C- ja C++-kieliin, ja se kehitettiinkin aikakaudella, jolloin C++ oli hallitsevassa asemassa ohjelmointikielenä [14]. Java suunniteltiin vetoamaan C++-kehittäjiin samankaltaisella syntaksilla, mutta tarjoamalla korkeamman tason kielenä ominaisuuksia, kuten esimerkiksi automaattisen muistinhallinnan, johon kuuluu myös roskienkeruu [14].

WORA (engl. *write once, run anywhere*), on Javan suunnittelua ohjannut filosofia. Yhdelle alustalle kirjoitettua Java-koodia voidaan ajaa millä tahansa muullakin alustalla ilman muutoksia koodiin. Koodia ajetaan virtuaalikoneessa, joka tulkkaa tavukoodia ajonaikaisesti alustariippuvaiseksi natiivikoodiksi.

Java ei ole puhdas oliokieli, sillä sen tyyppijärjestelmä sisältää luokkien lisäksi, tyhjäarvon (engl. *null type*) ja primitiivityyppejä. Tyypitykseltään Java on staattisesti tyypitetty, eli muuttujien tyypit tarkistetaan käännösaikana.

Kaikkien Javan funktioiden ja muuttujien täytyy olla määritelty luokan sisällä. Luokkien ja rajapintojen avulla Java hallitsee olio-ohjelmoinnin avainkonsepteja: abstraktiota, kapselointia, polymorfismia ja perintää.

Yksi Javan vahvuuksista on sen kattava *Collections*-sovelluskehys (engl. *framework*). Se sisältää yleisesti käytettyjä rajapintoja sekä niiden toteutuksia, joten sitä voidaan käyttää kirjaston tapaan.

2.2 Web-ohjelmointi

Javan yleistymistä auttoivat alkuvaiheessa suuresti sen edellisessä luvussa mainituista ominaisuuksista erityisesti siirrettävyys ja turvallisuus. Nämä olivat kaivattuja ominaisuuksia nopeasti yleistyvässä internetissä. [15]

Appletti on ohjelma, joka välitetään internetin välityksellä verkkoselaimelle, ja suoritetaan siinä ilman käyttäjän interaktiota. Applettien avulla on mahdollista siirtää toiminnallisuutta palvelimelta päätelaitteelle. Tämä oli mullistava uudistus internetin alkuaikoina, ja mahdollisti ensimmäistä kertaa dynaamisen tiedon välittämisen palvelimelta päätelaitteelle. Koska Appletit ovat Java-ohjelmia, niitä suoritetaan selaimen virtuaalikoneessa. Virtuaalikone ratkaisee siirrettävyysongelman, sillä sama ohjelma toimii siinä käyttöjärjestelmästä ja verkkoselaimesta huolimatta, olettaen, että selaimessa on JVM-tuki. Myös turvallisuusnäkökohta täyttyy, kun ohjelmia suoritetaan virtuaalikoneessa, sillä ohjelma ei pääse vaikuttamaan virtuaalikoneen ulkopuoliseen järjestelmään. [15]

Servletti on vastaavasti pieni ohjelma palvelimella, jolla luodaan dynaamista sisältöä tarjoiltavaksi päätelaitteille ja näin voidaan dynaamisesti laajentaa palvelimen toiminnallisuutta. Myös servletit ovat erittäin siirrettäviä palvelinalustojen välillä vaatien palvelimelta ainoastaan tuen JVM:lle ja servlet-säiliölle (engl. *servlet container*). [15]

Edellä kuvatut appletit ovat jo jääneet monelta osin historiaan internetin muovauduttua nykyisenlaiseksi. Appletit ovat jääneet ilman tukea älypuhelinkäyttöjärjestelmien selaimissa, mikä on johtanut niiden suosion yleiseen hiipumiseen, ja korvautumiseen muilla tekniikoilla. Palvelinpuolen rakenteet ovat monimutkaistuneet ohjelmistojen ja palveluiden kasvaessa, ja MVC (engl. *model-view-controller*) on nykyään suosittu malli komponenttien toteuttamiseen, ja siellä Java on edelleen vahvasti käytössä. [15] Lisäksi palvelinpuolelle on kehitetty lukuisia erilaisia Javaa tukevia ohjelmistokehyksiä.

2.3 Android

Android on Googlen kehittämä maailman suosituin mobiilikäyttöjärjestelmä. Sen pohjana on monen käyttäjän Linux-järjestelmä, joka mahdollistaa sovellusten suorituksen eristämisen toisistaan ajamalla niitä erillisinä prosesseina, jotka kukin kuuluvat eri käyttäjille. [11]

Java sopii hyvin mobiiliohjelmointiin. Kuten aiemmin mainittiin, yksi Javan tärkeimmistä suunnitteluperiaatteista oli koodin liikuteltavuus ja toimivuus erilaisilla alustoilla. Mobiililaitteet ovat hyvä esimerkki alustasta, jolla laitteistokirjo on valtava. Java-ohjelmoijia on myös maailmalla miljoonia, joten valitsemalla suosittu kieli avataan ovet valtavalle määrälle potentiaalisia kehittäjiä.

Android-ohjelmat kirjoitetaan Javalla, mutta niitä ei suoriteta JVM:ssä. Android-versioon 4.4 saakka ohjelmia ajettiin Androidin omalla Dalvik-virtuaalikoneella, mutta myöhemmissä versioissa ohjelmat käännetään natiivikoodiksi ja suoritetaan ART-ympäristössä (engl. *Android runtime*). Käännetty Java-tavukoodi käännetään ensin dex-tiedostoksi, minkä jälkeen se ja muut tarvittavat resurssit paketoidaan apk-tiedostoksi. Apk-tiedosto on paketoitu ohjelmatiedosto, joka on valmis asennettavaksi Android-laitteeseen. [11]

3. KOTLIN

Kotlin on 2010-lukulainen yleiskäyttöiseksi suunniteltu JVM-kieli. Kotlinin suunnittelussa on pyritty virtaviivaistamaan kielen rakennetta Javaan nähden, jotta käytännön ohjelmointityöstä tulisi tehokkaampaa ja turvallisempaa [8]. Kieltä kehittää ohjelmistotalo JetBrains, jonka tuotteeseen myös Android Studio -kehitysympäristön nykyinen versio perustuu. Kotlin soveltuu hyvin Android-kehitykseen, ja Google nosti sen vuonna 2017 yhdeksi virallisesti tuetuista Android-kielistä Javan ja C++:n rinnalle [16].

3.1 Perusteet

Yksi Kotlinin kantavista teemoista on hyvä integraatio Java-koodin kanssa, ja se onkin suunnattu erityisesti ympäristöihin, joissa ajetaan Java-koodia. Kotlin on kuitenkin monipuolinen kieli, ja se voidaan kääntää Javan lisäksi myös JavaScriptiksi tai natiiviksi binäärikoodiksi. Natiivi koodi on tarkoitettu ympäristöihin ja tilanteisiin, joissa virtuaalikone ei ole toivottu ratkaisu. Kotlin/Native-teknologialta löytyy tuki esimerkiksi iOS:lle, MacOS:lle, Androidille, Windowsille ja Linuxille. [9]

Kotlin kannustaa muuttumattoman datan käyttöön ja erotteleekin muuttujat muuttumattomiin, joiden avainsana on *val* ja muuttuviin, joiden avainsana on *var*. Lisäksi kirjastototeutukset on eroteltu muuttamisen salliviin ja muuttamattomiin. Kotlinin syntaksissa muuttujan tyyppi kirjoitetaan nimen jälkeen, ja puolipiste rivin päättymisen merkkinä ei ole pakollinen. [8]

Tyypitykseltään Kotlin on vahvasti ja staattisesti tyypitetty. Kääntäjä tarkastaa siis tyypityksen käännösaikana, ja kaikilla muuttujilla on oltava tyyppi. Tyyppiä ei aina kuitenkaan tarvitse antaa, jos kääntäjä pystyy sen koodista päättelemään, sillä kielestä löytyy tuki myös tyypinpäättelylle (engl. *type inference*). Koska Kotlin tukee käännöstä myös heikosti tyypitettyihin, tai tyypittämättömiin kieliin, on siinä myös tyyppi *dynamic*. Tämän tyyppisille muuttujille kääntäjä ei tee ollenkaan tyyppitarkistuksia tai tyhjäarvojen (engl. *null*) tarkistuksia. Kotlin ei tee ollenkaan automaattisia tyyppimuunnoksia, mutta kieli tarjoaa muutoksiin valmiita funktioita. [8]

3.2 Tyyppijärjestelmä

Kotlinin tyyppijärjestelmä tarjoaa useita uusia ominaisuuksia, joita ei löydy Javasta. Näitä koodin luotettavuuden parantamiseen tähtääviä ominaisuuksia ovat esimerkiksi niin tyhjättävät tyypit (engl. *nullable types*) kuin muuttumattomat kokoelmatkin (engl. *read-only collections*). [8]

3.2.1 Tyhjättävät tyypit

Muiden modernien ohjelmointikielten tapaan ajonaikaisten virheiden määrää on pyritty Kotlinissa rajoittamaan. Tyhjättävien tyyppien avulla kääntäjä voi huomata monia mahdollisia virheitä käännösaikana, vähentäen näin ajonaikaisten virheiden mahdollisuutta. Kotlinissa kääntäjä valvoo, että tyhjättävää tyyppiä olevan muuttujan arvo tarkistetaan ennen sen käyttämistä. Kääntäjä ilmoittaa virheestä myös silloin, jos muuttuja ei ole tyhjättävää tyyppiä, mutta siihen yritetään sijoittaa tyhjäarvo. Java ei tunne tyhjättävää tyyppiä, ja siinä tyhjäarvon lukemisesta aiheutuneet *NullPointerException*-virheet ovat yleisiä. Tyhjäarvojen tarkastamisen helpottamiseksi Kotlinista löytyy turvallisen kutsun (engl. *safe-call*) *?.*-operaattori, joka yhdistää metodin kutsumisen ja tyhjäarvon tarkistamisen yhdeksi operaatioksi. Turvallisen kutsun operaattoria käytettäessä metodia kutsutaan ainoastaan silloin, kun arvo ei ole tyhjä. [8]

Kotlin osaa hyödyntää Java-koodin yhteydessä tyhjäarvojen huomautuksia (engl. *annotations*), mikäli näitä on koodissa käytetty ja ottaa ne huomioon muunnettaessa Java-tyyppiä Kotlin-tyyppiin. Mikäli huomautuksia ei ole käytetty, muunnetaan Java-tyypit alustatyypeiksi (engl. platform *type*). Kääntäjä ei ota kantaa alustatyypin sisältämään arvoon, vaan sallii sille sekä tyhjän arvon tarkastukset että tarkastamatta jättämisen. Alustatyyppien yhteydessä vastuu siis siirretään käyttäjälle, jolloin niitä käytetään kuten Javan tyyppejä. [8]

3.2.2 Primitiivityypit

Kääretyypit (engl. *wrapper types*) ovat Javan mekanismi, jonka avulla primitiivityyppi voidaan tarvittaessa kapseloida olioksi. Kotlin ei tunne eroa primitiivityypin ja kääretyypin välillä, mikä vähentää erilaisten tyyppien tarvetta. Kotlinin numerotyypit muunnetaan Javan primitiivityypeiksi, jos mahdollista, ajonaikaisen suorituskyvyn maksimoimiseksi. [8] Nollattavaa tyyppiä olevat muuttujat muutetaan Javan kääretyypeiksi. Jos Javan primitiivityyppi luodaan Kotlin-koodissa, tulee siitä ei-tyhjättävää (engl. *non-nullable*) tyyppiä. [8]

3.2.3 Kantatyyppi Any

Kotlinin tyyppihierarkian huipulla on tyyppi *Any*, mutta se on ei-nollattava tyyppi. Tästä syystä nollattavia arvoja varten on käytettävissä *Any?*-tyyppi. Javassa *Object*-tyyppi on kaikkien olioiden kantatyyppi, mutta primitiivityypit eivät kuulu osaksi samaa hierarkiaa. Kotlinissa myös primitiivityypit kuuluvat samaan hierarkiaan, joten *Any* kaikkien tyyppien kantatyyppi. [8]

3.2.4 Tyypit Unit ja Nothing

Unit vastaa Javan tyyppiä *void*, ja sitä käytetään ilmaisemaan funktion paluuarvoa, jolla ei ole merkitystä. *Unit*-tyyppiä ei tarvitse eksplisiittisesti palauttaa funktiosta. Toisin kuin *void* Javassa, *unit* on oikea ainokainen (engl. *singleton*) luokka. [8]

Nothing on tyyppi funktioille, joiden ei odoteta palauttavan mitään. Tällaisia funktioita ovat esimerkiksi ikuinen silmukka tai funktio, joka heittää suunnitellusti poikkeuksen. *Nothing*-tyyppi ei voi sisältää arvoa. Erilliset *nothing* ja *unit* -tyypit parantavat koodin ymmärrettävyyttä. Javassa ei ole vastinetta Kotlinin *nothing*-tyypille. [8]

3.2.5 Kokoelmat

Kotlinin kokoelmissa on Javaan verrattuna se merkittävä ero, että niissä on erilliset rajapinnat datan lukemiselle sekä muuttamiselle. Tämä parantaa koodin luettavuutta, ja on suositeltavaa käyttää pelkän lukemisen sallivia rajapintoja, jos se vain on mahdollista. Jos muuttujaan on useita viitteitä, sen muuttumattomuutta ei voida taata, vaikka sitä käsiteltäisiin lukemisen sallivan rajapinnan kautta, sillä muut viitteet voivat käyttää muuttamisen sallivaa rajapintaa. [8]

Kaikki Kotlinin kokoelmat ovat Javan kokoelmarajapintojen instansseja, joten kääreitä, datan kopioimista tai muuta muokkaamista ei tarvita kielten välillä. Vaikka käytetty kokoelma olisi Kotlin-koodissa pelkästään lukemisen salliva, on sen muokkaaminen silti mahdollista Java-koodin puolella, koska Javassa kaikki kokoelmat ovat muokkaamisen sallivia. [8]

Laajennettaessa tai korvattaessa Java-kokoelmaan kuuluvaa metodia (engl. *method*) Kotlin-koodissa on mahdollista vaikuttaa kokoelman muokattavuuteen, nollattavuuteen sekä sen alkioiden nollattavuuteen. Se mahdollistaa Javaa suuremman kontrollin metodin käytössä. [8]

3.3 Luokat ja oliot

Kotlinissa luokat ovat oletuksena lopullisia (engl. *final*), toisin kuin Javassa, jossa ne ovat avoimia (engl. *open*). Lopullisuus tarkoittaa sitä, että periytyvät luokat eivät voi uudelleen kirjoittaa isäntäluokan metodin toteutusta. Avoimet (engl. *open*) luokat ovat teoriassa käteviä, mutta voivat käytännössä johtaa särkyvän yliluokan (engl. *fragile base class*) ongelmaan. Ongelma on siinä, että muutokset kantaluokkaan voivat helposti aiheuttaa odottamattomia muutoksia periytetyissä luokissa. Näin voi tapahtua, jos periytetyt luokat ovat muokanneet kantaluokan metodien toteutuksia tavoilla, joita kantaluokan tekijä ei ole ennakoinut. Avoimet luokat ovat myös Kotlinissa mahdollisia, mutta ne täytyy erikseen määrittää sellaisiksi. [8]

Näkyvyystasoltaan Kotlinin luokat ovat oletuksena julkisia. Javan oletusnäkyvyystaso on pakkaus (engl. *package*), joka tarkoittaa näkyvyyttä saman pakkauksen sisällä, jollaista Kotlin ei tunne lainkaan. Kotlin käyttää pakkauksia ainoastaan koodin järjestämisessä eri nimiavaruuksiin, ei Javan tapaan näkyvyyden hallintaan. Kotlinissa määreen *internal* voidaan nähdä tarjoavan Javaa aidomman tavan kapselointiin, sillä se tarkoittaa näkyvyyttä samalla kerralla käännettävien tiedostojen, moduulin kesken. [8]

Luokalle, jota käytetään tiedon säilömiseen, on usein tarve tehdä tiedon käsittelyä helpottavia metodeja. Tällaisia metodeja ovat esimerkiksi luokan nimen merkkijonoksi muuttaminen, luokan ilmentymien kesken vertailu ja tiivisteen generoiminen hajautustaulussa säilömistä varten. Kotlinin dataluokille (engl. *data classes*) nämä metodit generoidaan automaattisesti. Tarjoamalla myös kopiointimetodin, Kotlin kannustaa tekemään muuttujista muuttumattomia. Dataluokat ovat yksi Kotlinin tarjoamista tavoista *boilerplate*-koodin, eli monessa paikassa toistuvan koodin vähentämiseen, mikä tekee koodista suoraviivaisempaa. [8]

Ainokaisen (engl. *singleton*) eli luokan, jolla täsmälleen yksi ilmentymä, luomisen helpottamista varten Kotlin tarjoaa avainsanan *object*. *Object* yhdistää sekä luokan että sitä ilmentävän olion luomisen yhteen esittelyyn. [8]

3.4 Funktiot

Funktioiden ja rajapintojen suhteen Kotlin ei koita keksiä pyörää uudelleen, vaan se hyödyntää Javan *Collections*-sovelluskehystä. Tämä mahdollistaa paremman Java- ja Kotlin-koodin yhdistelemisen, sillä Kotlin-koodista voidaan kutsua Javan funktioita ja toisinpäin. Argumenttien nimeäminen funktiokutsuissa, funktion parametrien oletusarvot ja laajennettavat funktiot ovat kaikki Kotlinin tukemia ominaisuuksia, joita Javasta ei löydy. [8]

Javassa, kuten muissakin olio-ohjelmointikielissä, kaikki funktiot ovat osana luokkia. Javasta poiketen Kotlin tukee korkeimman tason funktioita eli funktioita, jotka eivät ole minkään luokan, olion tai rajapinnan jäseniä. Korkeimman luokan funktiot määritellään siis suoraan kooditiedoston ylimmällä tasolla. Samalla tavalla myös muuttujien määrittäminen korkeimmalla tasolla on mahdollista. [8]

Laajennusmetodit (engl. *extension functions*) ovat tapa, jonka avulla Kotlin mahdollistaa uuden toiminnallisuuden tarjoamisen jo olemassa oleviin luokkiin ja kirjastoihin. Laajennusmetodit tarkoittavat yksinkertaisuudessaan luokan metodeja, jotka on määritelty luokan määrittelyn ulkopuolella. Laajennettavan luokan ei tarvitse olla edes kirjoitettu samalla kielellä kuin laajennusmetodin, kunhan ne käännetään ennen ajoa samalle kielelle. Esimerkiksi luokka voisi olla kirjoitettu jollain muulla JVM-kielellä, mutta laajennusfunktio Kotlinilla. Laajennusmetodit ovat myös tapa, jolla Kotlin tarjoaa laajemman ra-

japinnan Javan *Collections*-sovelluskehykseen. Laajentamalla jo olemassa olevia kirjastoja yhteensopivuus säilyy, mutta samalla mahdollistetaan lisätoiminnallisuuden tarjoaminen. [8]

3.5 Asynkroninen ohjelmointi

Vuorottaisrutiinit (engl. *coroutine*) ovat Kotlinin käyttämä ratkaisu asynkroniseen ohjelmointiin. Kotlinin toteutus mahdollistaa asynkronisen ohjelmakoodin kirjoittamisen ilman erityissyntaksia. Kutsuvassa koodissa käytetään avainsanaa *launch*, joka sananmukaisesti käynnistää vuorottaisrutiinin, ja normaalista funktiosta saadaan vuorottaisrutiini laittamalla sen eteen avainsana *suspend*. Vuorottaisrutiinit ovat hyödyllisiä esimerkiksi suoritettaessa raskaita operaatioita, jolloin koko ohjelman suorituksen ei tarvitse jäädä odottamaan näiden operaatioiden suorituksen valmistumista. Pääsäie voi jatkaa eteenpäin heti vuorottaisrutiinin kutsumisen jälkeen, ja vuorottaisrutiini palauttaa tuloksensa pääsäikeelle myöhemmin saadessaan suorituksensa loppuun. Kotlinin vuorottaisrutiinit on toteutettu pääosin kirjastojen avulla. [10]

3.6 Monialustaohjelmointi

Kuten aiemmin mainittiin, tukee Kotlin monia erilaisia alustoja. Kotlinin versioissa 1.2 ja 1.3 on uutena ja kokeellisena ominaisuutena tuki monialustaohjelmoinnille (engl. *multiplatform programming*). Tällä tarkoitetaan koodin jakamista ja uudelleenkäyttämistä eri alustojen välillä määrittelemällä yhteinen rajapinta kutsuille, mutta luomalla rajapinnalle alustakohtaiset toteutukset. Alustakohtaiset toteutukset hyödyntävät alustan omia kirjastoja. Monille yleisesti käytetyille toiminnoille, kuten HTTP-kutsuille on kirjastototeutukset useissa kielissä, joita voidaan myös hyödyntää alustakohtaisissa toteutuksissa. Yksi käyttötapa monialustaisuudelle Kotlinissa on korkean tason toiminnallisuuteen ja sovelluksen toimintalogiikkaan liittyvän koodin jakaminen iOS- ja Android-sovellusten kesken. [9]

3.7 Android-tuki

Jo ennen Google I/O 2017 -tapahtumaa, jossa julkistettiin [16] Kotlinin uusi asema yhtenä virallisista Android-kielistä, Kotlin vaikutti varteenotettavalta kieleltä Android-sovelluskehityksessä käytettäväksi. Siinä oli monia Javasta puuttuvia moderneja ominaisuuksia, mutta se säilytti kuitenkin täyden yhteensopivuuden Javan kanssa. [8] Myös työkalutuki Kotlinin puolelta oli kunnossa jo ennen julkistusta, sillä Android-studio, joka on Googlen tarjoama virallinen IDE (engl. *integrated development environment*) Android-kehitykseen, on Kotlinin kehittämän JetBrains-yrityksen käsialaa. Virallinen tuki Android-kehitykseen on tuonut mukanaan uusia kannustimia Kotlinin käyttöön, kuten lisätyn näkyvyyden, sekä Kotlinin huomioimisen Android SDK:ssa (engl. *software development kit*). [5]

”Kirjoita parempia Android-sovelluksia nopeammin.” Näin kirjoitetaan Googlen Android-kehittäjille suunnatun sivuston Kotlin-osiossa. Androidin versiossa 9, eli API-versiossa 28, Androidiin on lisätty aiemmissa luvuissa mainitut tyhjäarvojen huomautukset, jotta Kotlinin Androidiin tuomia uusia ominaisuuksia voitaisiin hyödyntää entistä paremmin. Lisäksi Androidin rajapintadokumentaatio on saatavilla myös Kotlinille kirjoitettuna. Sivustolla on myös useita linkkejä resursseihin, joiden avulla kehittäjät voivat tutustua Kotliniin. [5]

Kotlin on yhteensopiva myös vanhempien Android-versioiden kanssa johtuen JDK (engl. *Java development kit*) 6 -yhteensopivuudesta. Kattava Java-yhteensopivuus tukee Kotlinin tavoitetta olla helposti lähestyttävä kieli Java-kehittäjille, ja siirtymisen voi tehdä vähän kerrallaan, ilman tarvetta kirjoittaa koko ohjelmistoa kerralla uudelleen. Kotlinin käytöllä ei ole heikentävää vaikutusta sovelluksen suorituskykyyn, sillä sen tavukoodin rakenne on hyvin lähellä Javan tavukoodia. Lisäävää (engl. *incremental*) kääntämistä käyttämällä Kotlin-koodin käännösajat saadaan jopa Javan käännösaikoja nopeammiksi. Lisäksi käännetyn tiedostonkaan koossa ei ole suurta eroa Kotlinin kompaktin ajonaikaisen kirjaston vuoksi. [9]

3.8 Palvelinohjelmointi

Toinen merkittävä käyttökohde Kotlinille Androidin lisäksi on palvelinohjelmointi. Palvelinohjelmointiin kuuluu niin HTML-sivun selaimelle palauttavia sovelluksia, mobiilisovellusten taustaohjelmia (engl. *backend*) kuin myös monia muita sovelluksia. Javalla on pitkä historia tällä sovellusalueella, joten sille on ehditty tekemään paljon sovelluskehyksiä ja muita työkaluja, joita palvelinohjelmoinnissa usein hyödynnetään. Sovellusalueella on myös yleistä, että laajennetaan jo olemassa olevaa järjestelmää sen sijaan, että tehtäisiin jotain täysin uutta. [8]

Kuten Androidinkin kanssa, Kotlin hyötyy suuresti vahvasta Java-yhteensopivuudestaan palvelinohjelmoinnissa. Kotlinin avulla on vaivatonta ensin kokeilla kieltä esimerkiksi uuden komponentin kanssa ilman, että vanhaa koodipohjaa tarvitsisi lähteä muokkaamaan. Kotlinin kehitysympäristöstä, IntelliJ IDEA:sta, löytyy myös hyvä tuki lisäosille esimerkiksi sovelluskehyksien osalta. Muun muassa Spring Framework, yksi suosituimmista Javan palvelinohjelmoinnin sovelluskehyksistä, tukee virallisesti Kotlinia, hyödyntäen esimerkiksi aiemmin mainittuja tyhjättäviä tyyppejä. [9]

4. KOTLININ SUOSIO

Tässä luvussa käsitellään Kotlinin suosiota ja pohditaan sen syitä. Ensin käsitellään sen suosiota erilaisten mittareiden avulla.

4.1 Suosio erilaisilla mittareilla mitattuna

Ohjelmointikielen suosiota voi mitata monella tavalla. Yksi mittari voi mahdollisesti olla sitä käyttävien kehittäjien määrä. Ongelmana on, että etenkään työkseen ohjelmointia tekevän ei ole aina mahdollista vaikuttaa käytettävään ohjelmointikieleen, sillä kieli voi olla projektin tai työnantajan sanelema. Työnantajan näkökulmasta kannattaa taas suosia kieltä, jota osaavaa työvoimaa on hyvin saatavilla, ja joka on jo todettu tuotantovalmiiksi. Tämä johtuu siitä, että turhia riskejä teknologiavalinnoissa pyritään välttämään. Ohjelmoija voi taas vapaa-ajallaan suosia aivan erilaisia kieliä, kun tavoitteena ei ole maksimoida koodin kaupallista potentiaalia, vaan toteuttaa henkilökohtaisia mielihaluja. Tässä luvussa käydään läpi Kotlinin suosiota useilla erilaisilla mittaustavoilla mitattuna, jotta kielen suosiosta saadaan muodostettua kokonaisvaltaisempi kuva.

4.1.1 Kyselyt

Stack Overflow on vuonna 2008 perustettu, nykyisin maailman suosituin ohjelmointiin keskittyvä internetsivusto, jolla on yli 50 miljoonaa kuukausittaista käyttäjää. Stack Overflow keskittyy ohjelmointiaiheisten ongelmien ratkaisuun, ja se tarjoaa kattavan foorumin, jolla käyttäjät voivat sekä kysyä ohjelmointiaiheisia kysymyksiä että vastata muiden käyttäjien jättämiin kysymyksiin. [17] HackerRank taas on vuonna 2012 perustettu ohjelmistoalan työnhakuun keskittyvä alusta, jonka tavoitteena on saattaa yhteen ohjelmistoalan työnantajat sekä työnhakijat [7]. Stack Overflow- ja HackerRank- sivustot ovat kumpikin tahoillaan suorittaneet kattavat kyselyt käyttäjäkunnalleen.

Stack Overflow -sivuston Developer Survey 2018 -kyselyyn vastasi yli 100 000 käyttäjää 183 maasta. Kotlin oli kyselyssä mukana ensimmäistä kertaa, sijoittuen suosituimpien ohjelmointikielten listauksessa 22. sijalle, kun kysyttiin, mitä kieliä vastaajat olivat käyttäneet: sitä oli käyttänyt 4,5 % vastanneista. Javaa käyttäneitä oli puolestaan 45,3 % vastanneista, mikä oikeutti sen listan sijaan 5. Kysymykseen oli vastannut 78 334 käyttäjää. Rakastetuimpien kielten listauksessa vastaajilta kysyttiin kiinnostuksesta jatkaa kielellä ohjelmointia, liittyen niihin kieliin, joita hän oli aiemmin vastannut käyttävänsä. Kotlin sijoittui listauksessa toiseksi, kun kolme neljästä Kotlinin käyttäjästä ilmoitti kiinnostuksestaan jatkaa kielen parissa – Javaan tyytyväisiä oli ainoastaan joka toinen. [18] Rakastetuin kieli valittiin siis kieleen tyytyväisten suhteellisen osuuden, ei lukumäärän perus-

teella. Vaikka Javaan tyytyväisiä oli suhteellisesti vähemmän, oli heitä kuitenkin määrällisesti selvästi enemmän. Kysyttäessä kieliä, joita vastaajat olivat kiinnostuneet kokeilemaan, 12,4 % valitsi Kotlinin. Javan oli valinnut 10,5 %, joka vastasi listalla sijaa 6, kun Kotlin oli neljäs [18]. Javan kokeilemisesta kiinnostuneiden vähäisempää määrää voi selittää se, että lähes puolet kyselyyn vastanneista oli jo käyttänyt sitä.

HackerRank-sivuston Developer Skills -kysely keräsi yhteensä 39 441 vastausta. Yhtenä kysymyksistä kyselyssä oli, mitä kieliä vastaajat aikoivat opetella seuraavaksi. Kotlin oli 4. yleisin vastaus, yli neljänneksen kyselyn vastaajista ollessa kiinnostunut siitä, kun Java kiinnosti ainoastaan joka kymmenettä vastaajaa. [7] Myös HackerRankin kyselyn vastauksiin voi vaikuttaa se, että Java on jo monien kehittäjien ennestään käyttämä kieli, jolloin sille on vähemmän uusia mahdollisia käyttäjiä.

4.1.2 Hakukoneet

TIOBE Index on kuukausittain päivittyvä listaus, joka järjestää 50 eniten hakukoneosumia kerännyttä ohjelmointikieltä löydettyjen sivujen määrän mukaiseen järjestykseen. Tiedot haetaan Alexa Internetin [2] mukaan 25 käytetyimmällä hakukoneella. Tilastoitava haku muodostuu 2 sanasta, joista ensimmäinen on käytetyn ohjelmointikielen nimi ja toinen sana on ”programming”. TIOBE Index siis laskee, kuinka monta ohjelmointikieleen liittyvää verkkosivua hakukoneilla löydetään. Lisäksi kielille lasketaan niitä koskevien sivujen määrä prosentteina kaikkien listattavien kielten sivujen yhteismäärästä. [22] Joulukuun 2018 TIOBE Index -listauksessa Java on sijalla 1, kattaen lähes 16 % kokonaissivumäärästä, joka on 2,66 prosenttiyksikköä enemmän kuin vuotta aiemmin. Kotlin löytyy listan sijalta 36, kattaen ainoastaan 0,253 % siivun kokonaissivumäärästä. [21]

PYPL PopularitY of Programming Language on toinen kuukausittainen hakukonetuloksien pohjalta tehty listaus. Sen mukaan TIOBE Index -listaus ei anna oikeanlaista kuvaa ohjelmointikielten suosiosta, sillä vanhalla ja kuolleellakin kielellä voi edelleen olla paljon verkossa olevaa materiaalia, vaikka sen käyttäjämäärät olisivat pudonneet vähäisiksi. Moitteita saa myös hakulauseke, sillä sen sanotaan olevan huonosti valittu ja vääristävän tuloksia, sillä sanan ”programming” lisäämisen vaikutus hakutulokseen riippuu paljon kielestä, ja se jätetään monesti pois. PYPL mittaa listauksessaan kielten kurssien (engl. *tutorial*) Google-hakujen kuukausittaisia määriä, ja järjestää 23 haetuinta kieltä hakumäärien mukaiseen järjestykseen. Java on joulukuun 2018 listauksen mukaan 2. haetuin kieli 21,56 % osuudella, reilun prosenttiyksikön laskulla vuotta edeltävään hakumäärään. Kotlin on listan sijalla 16, ja se keräsi 1,12 % kaikista hauista, nousten 0,3 prosenttiyksikköä vuoden 2017 joulukuun listaukseen verrattuna. [4]

4.1.3 GitHub-ohjelmavarastot

GitHub on maailman suosituin Git-versionhallintaa käyttävien ohjelmistoprojektien säilytyspaikka internetissä: sivusto tarjoaa kodin yli 100 miljoonalle ohjelmavarastolle (engl. *repository*) [6]. GitHub julkaisee vuosittain The State of the Octoverse -raportin perustuen dataan, joka on kerätty sivustolla säilöttävistä ohjelmavarastoista. Vuoden 2018 raportissa Kotlin sijoittui eniten suosiotaan kasvattaneiden kielten listan kärkipaikalle. Listauksessa vertailtiin kehittäjämäärien muutosta vuoden takaiseen ohjelmavarastoissa, jotka oli merkitty kyseisen kielen tunnisteella (engl. *tag*). Lisäksi raportissa huomioitiin, että staattisesti tyypitetyt, tyyppiturvallisuuteen ja yhteentoimivuuteen (engl. *interoperability*) panostavat ohjelmointikielet kasvattivat suosiotaan voimakkaasti. Kotlin oli yksi näistä kielistä 2,6-kertaistaen suosionsa. [20] Raportissa ei otettu kantaa kielellä kirjoitetun koodin suhteelliseen osuuteen ohjelmavarastossa, koska laskettiin ainoastaan tunnisteiden esiintymismääriä. Samassa ohjelmavarastossa voi olla useammalla eri ohjelmointikielellä kirjoitettua koodia, jolloin se merkitään kaikkien niiden tunnisteilla.

4.1.4 Kysyntä työmarkkinoilla

TrendySkills-sivusto suodattaa ohjelmistoalan työpaikkailmoituksia suosituilta sivustoilta 14 eri maasta, ja tarjoaa näihin ilmoituksiin liittyvää tilastotietoa. Tarkasteltaessa tilastoa, joka kuvaa eri ohjelmointikielten osaajien avoimien työpaikkojen määrää aikavälillä 19.12.2017–19.12.2018 kaikista 14 maasta, Java-kehittäjiä haetaan kaikkein eniten, 21 prosentissa kaikista ilmoituksista. Kotlin ei mahtunut mukaan listattujen 10 kysytyimmän kielen joukkoon. [23]

Indeed on yli 200 miljoonalla kuukausittaisella käyttäjällään maailman suosituin työpaikkailmoitusten hakemiseen tarkoitettu hakukone [1]. Coding Dojo -sivusto on blogissaan analysoinut Indeed.com:n kautta 17.11.2017 löytyneitä ohjelmistoalan työpaikkailmoituksia, ja listannut 7 ohjelmointikieltä, joiden kehittäjistä oli tuolloin suurin kysyntä. Myös tämän vertailun kärjessä oli Java, 62 000 avoimella työpaikalla. Kotlin ei ollut edustettuna tämänkään vertailun lopputuloksissa. [12]

Jo aiemmin mainitussa HackerRank-sivuston Developer Skills -kyselyssä kysyttiin myös työnantajilta, minkä ohjelmointikielen osaajia he hakevat. JavaScript ja Java pitivät kärkipaikkaa: molempien kielten osaajia etsi yli 47 % työnantajista. Kotlin ei ollut 15 muun listatun kielen joukossa. [7]

4.2 Suosion syyt

Kotlinin Stack Overflow Developer Survey 2018 -menestyksen [18] innoittamana ohjelmistoyritys Pusher toteutti kyselyn, jolla pyrittiin selvittämään syitä Kotlinin äänestämiselle toiseksi rakastetuimmaksi ohjelmointikieleksi. Kysely keräsi tammikuusta maaliskuuhun 2018 yhteensä 2744 vastausta. [19]

Noin 80 % yrityksistä, joissa vastanneet työskentelivät, käytettiin Kotlinia. Valtaosassa yrityksistä Kotlinia käytti työssään vain 1–5 henkilöä. Ainoastaan 3,7 % ilmoitti Kotlinia käyttävien määrän olevan yli 50 henkilöä. Eniten vastauksia kerännyt yksittäinen vastausvaihtoehto oli 2–5 henkilöä, jonka huomioitiin myös olevan sama kuin keskimääräisen Android-kehitystiimin koko. [19]

Kyselyn perusteella Kotlinin suosio kasvoi aluksi hitaasti, mutta kuitenkin kaksinkertaistuen vuosittain. Ensimmäinen iso hyppäys käyttäjämäärissä osuu vuodelle 2015, jolloin käyttäjämäärä vastaajien keskuudessa nousi 1,4 prosentista 7,7 prosenttiin. Vuonna 2016 käyttäjiä oli jo 19,5 prosenttia. [19] Raportin mukaan Jake Whartonin vuoden 2015 alussa kirjoittamalla analyysillä, Using Project Kotlin for Android [24], oli suuri vaikutus suosion nopeaan kasvuun kyseisenä aikana. Wharton työskenteli tuolloin Squarella, joka on tunnettu suosituista avoimen lähdekoodin Android-kirjastoistaan. Seuraava suuri hyppäys käyttäjämäärissä tapahtui vuoden 2017 toukokuussa järjestetyn Google I/O -tapahtuman jälkeen, jossa julkistettiin Kotlinin nostaminen viralliseksi Android-kieleksi: kielen kokonaiskäyttäjämäärä kaksinkertaistui seuraavan puolen vuoden aikana. Vuoteen 2016 saakka käyttäjäkunta koostui enimmäkseen ohjelmistoalan ammattilaisista, mutta vuonna 2017 suosio opiskelijoiden keskuudessa kasvoi räjähdysmäisesti nousten 9 prosentista yli 86 prosenttiin. [19]

Kysyttäessä kehittäjien taustoja, yleisimpiä mainittuja ohjelmointikieliä olivat Java ja JavaScript, Javan ollessa selvästi yleisin. Tämä ei ole yllättävää, sillä lähes 80 % ilmoitti käyttävänsä Kotlinia juuri Android-kehitykseen, jota oli aiemmin tehty Javalla. Toiseksi yleisimpiä käyttökohteita olivat palvelinpään sekä kirjastojen ohjelmointi, joita kehittäjistä vajaa kolmannes ilmoitti tekevänsä. [19] Lisäksi palvelinpäässä Java on yleisesti käytössä. [15] Toisaalta myös JavaScriptin käyttö palvelinpäässä on nykyään erittäin suositu ratkaisu, etenkin Node.js-sovelluskehyksen suosion myötä [18]. Kotlinin virallinen internetsivusto [9] oli selvästi suosituin lähdemateriaali kielen opiskeluun, ja yli puolet kyselyyn vastanneista oli käyttänyt sitä [19].

Jos Kotlinia käytettiin työprojektissa, oli paljon yleisempää, että vain osassa projektia käytettiin sitä. Kun kokonaan Kotlinilla kirjoitettujen projektien osuus henkilökohtaisista projekteista oli yli puolet, ainoastaan reilulla neljänneksellä koko työprojekti koostui Kotlinista. Kotlinin käyttö oli yleisempää pienemmissä, helpommin muutettavissa ja kokeellisemmissa projekteissa. Vastaajista 87 % oli tehnyt koodin migraatioita Javasta Kotliniin, mutta yli neljänneksen piti peruuttaa muutokset. Osa Javaan palaamisen syistä oli teknisiä, osa työorganisaatioon liittyviä. Tekniset syyt liittyivät vastausten mukaan usein heijastuksia (engl. *reflection*) käyttäviin tai koodia generoiviin työkaluihin. [19] Heijastus on koodin ajonaikaisen tarkastelun mahdollistava rajapinta Javassa [15]. Ainoastaan noin neljännes oli kääntänyt Kotlinia muuksi kuin Javan tavukoodiksi: 15,5 prosenttia natiiviksi koodiksi, 11,7 prosenttia JavaScriptiksi [19].

Vastaajien keskuudessa Kotlinin suosikkiominaisuudeksi valikoitui selvällä erolla tyhjäarvoturvallisuus (engl. *null safety*). Kielen muut pidetyimmät ominaisuudet, joista monia käsiteltiin tämän työn aiemmissa luvuissa, olivat järjestyksessä seuraavat: laajennusmetodit, Java-yhteentoimivuus, dataluokat, korkean tason funktiot, tyypinpäättely, monialustaohjelmointi ja vuorottaisrutiinit. [19]

5. YHTEENVETO

Tässä työssä tutkittiin Kotlinia ohjelmointikielenä. Tarkoituksena oli selvittää sen suosion kasvua ja syitä. Kotlinin ominaisuuksia lähdettiin avaamaan erityisesti vertailemalla sitä Javaan, ja kuinka se eroaa siitä, koska JVM-kielenä se on suuntautunut vahvasti samalle sovellusalueelle. Teoriaosuudessa käytiin ensin läpi perustietoa molemmista kielistä, jonka jälkeen esiteltiin Kotlinin Javasta erottavia uusia ominaisuuksia.

Teoriaosuuden jälkeen käytiin läpi Kotlinin suosiota. Suosion mittaamisessa käytettiin jo valmiiksi internetissä julkisesti saatavilla olevaa materiaalia. Mukaan valikoitui kattavia ohjelmointiaiheisia kyselyitä, hakukoneiden dataa hyödyntäviä tilastoja, ohjelmavarastoista saatua tilastodataa ja työpaikkailmoituksista saatua tilastodataa.

Kotlinin käyttäjämäärät ovat edelleen verrattain pieniä etenkin Javaan verrattuna. Javalle on vuosien saatossa syntynyt vahva asema tietyillä sovellusalueilla, kuten Android-ohjelmoinnissa ja web-ohjelmoinnissa, eikä se ole helposti horjutettavissa. Java on yksi maailman eniten käytetyistä ohjelmointikielistä, joten sen ympärille on kehitetty paljon erilaisia teknologioita ja sillä on paljon osaajia ympäri maailman. Toisaalta Javan yleisyys luo Kotlinille valtavan potentiaalin, koska mahdollisia käyttökohteita ja käyttäjiä on todella paljon. Virallisen Android-kielen asema on selvästi kiihdyttänyt Kotlinin yleistymistä, ja se on synnyttänyt kielen ympärille paljon kuhinaa. Googlen vahvan tuen ja kehittäjäyhteisön innostuksen perusteella sen tulevaisuus Android-kielenä vaikuttaa olevan vahvalla pohjalla, ja juuri Androidin kanssa sen käyttö onkin tällä hetkellä yleisintä. Toisaalta työmarkkinoilla kielellä on vielä vähän kysyntää, toisin kuin Javalla.

Työssä onnistuttiin muodostamaan melko kattava kuva Kotlinin suosion kasvusta tähän päivään mennessä. Sen suosioon liittyvää materiaalia oli runsaasti, mutta sitä oli lähinnä virallisen Android-tuen jälkeiseltä ajalta. Kokonaisvaltaisemman kuvan muodostamiseksi olisi ollut hyvä, jos tietoa olisi enemmän myös aiemmilta vuosilta. Suosion analysoimiseen käytetyt lähteet mittasivat suosiota erilaisista näkökulmista, joten niiden avulla voitiin tarkastella erilaisia suosioon vaikuttavia tekijöitä. Kotlinia koskeva taustaluku pohjautuu paljolti yhteen lähteeseen, mikä ei ole parhaiden käytäntöjen mukaista. Tämä johtuu kuitenkin siitä, että Kotlin on melko tuore ohjelmointikieli, ei siitä julkaistua painettua materiaalia ole vielä kovin laajalti saatavilla.

Kotlin on saanut innokkaan vastaanoton etenkin Android-kehittäjien keskuudessa. Vahvimmat edellytykset sen käytön yleistymiselle ovatkin juuri Android-puolella, jossa sillä on jo hyvä tuki sekä kehitysympäristön että itse Android-alustan puolelta. Mielenkiintoinen jatkotutkimuksen aihe voisi olla Kotliniin siirtymiseen estävien tekijöiden selvittäminen. Tämä voisi auttaa selvittämään sen yleistymispotentiaalia perusteellisemmin.

LÄHTEET

[1] About Indeed, Saatavissa (viitattu 20.12.2018): https://www.indeed.com/about

[2] Alexa Internet, Saatavissa (viitattu 18.12.2018): https://www.alexa.com/

[3] B. Baesens, A. Backiel, S.V. Broucke, Beginning Java Programming: The Object-Oriented Approach, Wrox, Somerset, 2015, 672 p.

[4] P. Carbonnelle, PYPL PopularitY of Programming Language Index, Saatavissa (viitattu 17.12.2018): http://pypl.github.io/PYPL.html

[5] Develop Android apps with Kotlin, Saatavissa (viitattu 18.12.2018): https://developer.android.com/kotlin/

[6] GitHub is how people build software, Saatavissa (viitattu 19.12.2018): https://github.com/about

[7] HackerRank 2018 Developer Skills Report, Saatavissa (viitattu 17.12.2018): https://research.hackerrank.com/developer-skills/2018

[8] D. Jemerov, S. Isakova, Kotlin in Action, 1st ed. Manning Publications, 2017, 360 p.

[9] Kotlin Language Reference, Saatavissa (viitattu 16.12.2018): https://kotlinlang.org/docs/reference/

[10] Kotlin Language Tutorials, Saatavissa (viitattu 16.12.2018): https://kotlinlang.org/docs/tutorials/

[11] B. Kurniawan, Java for Android, 2nd ed. Brainy Software Inc, Brossard, Quebec, Canada, 2015, 712 p.

[12] S. Misirlakis, The 7 Most In-Demand Programming Languages of 2018, Saatavissa (viitattu 17.12.2018): https://www.codingdojo.com/blog/7-most-in-demand-programming-languages-of-2018/

[13] Mobile Operating System Market Share Worldwide, Saatavissa (viitattu 18.12.2018): http://gs.statcounter.com/os-market-share/mobile/worldwide

[14] P. Sarang, Java Programming, Oracle Press, New York, 2012, 672 p.

[15] H. Schildt, Java The Complete Reference, 7th ed. McGraw-Hill Publishing, 2006, 1024 p.

[16] M. Shafirov, Kotlin on Android. Now official, Saatavissa (viitattu 18.12.2018): https://blog.jetbrains.com/kotlin/2017/05/kotlin-on-android-now-official/

[17] Stack Overflow, Saatavissa (viitattu 17.12.2018): https://stackoverflow.com/

[18] Stack Overflow Developer Survey Results 2018, Saatavissa (viitattu 17.12.2018): https://insights.stackoverflow.com/survey/2018/

[19] The State of Kotlin 2018, Saatavissa (viitattu 20.12.2018): https://pusher.com/state-of-kotlin

[20] The State of the Octoverse 2018, Saatavissa (viitattu 17.12.2018): https://octoverse.github.com/projects

[21] TIOBE Index for December 2018, Saatavissa (viitattu 17.12.2018): https://www.tiobe.com/tiobe-index/

[22] TIOBE Programming Community Index Definition, Saatavissa (viitattu 18.12.2018): https://www.tiobe.com/tiobe-index/programming-languages-definition/

[23] Trends for Languages, Saatavissa (viitattu 17.12.2018): https://trendyskills.com/

[24] J. Wharton, Using Project Kotlin for Android, Saatavissa (viitattu 17.12.2018): https://docs.google.com/document/d/1ReS3ep-hjxWA8kZi0YqDbEhCqTt29hG8P44aA9W0DM8